

Application No.: 09/620,102

Docket No.: 00-VE20.57

REMARKS

Claims 1-19 are pending. In the Office Action, claims 1-6 are rejected under 35 U.S.C. § 103(a) as obvious over US 6,629,128 ("Glass") in view of US 6,510,550 ("Hightower"), and further in view Pospisil et al., *On Performance of Enterprise Java Beans* ("Pospisil"). Claims 7-19 are rejected under 35 U.S.C. § 103(a) as obvious over Glass in view of Pospisil. For the reasons set forth below, applicants respectfully traverse these rejections.

Claims 1, 7, and 10 have been amended by way of this response. The amendment to claim 1 is intended to more clearly recite the claimed invention. The present amendment to claim 1 in no way alters the scope of the claim. The amendment to claim 7 adds a single comma in line 2 and is intended solely to correct a grammatical omission. The present amendment to claim 7 in no way alters the scope of the claim. Similarly, the amendment to claim 10 is intended to correct a syntactical irregularity and a redundancy. The present amendment to claim 10 in no way alters the scope of the claim.

I. REJECTION OF CLAIMS 1-6 AS OBVIOUS OVER GLASS IN VIEW OF HIGHTOWER AND FURTHER IN VIEW OF POSPISIL

Claims 1-6 are rejected as obvious over Glass in view of Hightower and Pospisil. However, the Examiner has failed to state a *prima facie* case of obviousness at least because (1) Glass, Hightower, and Pospisil, even when combined, fail to teach or suggest each and every one of the limitations of Applicants' claims, and (2) there is no teaching or suggestion in the prior art references to combine them to achieve the claimed invention. See MPEP § 2143. Accordingly, Applicants respectfully traverse the rejections of claim 1-6 for the independent reasons set forth below.

A. Claim 1: "returning . . . a handle to a functional bean"

Claim 1 recites, *interalia*, "returning to said client program a handle to a functional bean appropriate to the client request, wherein the functional bean is configured to model a business function." The Examiner asserts that Glass teaches this claim limitation. (Office Action, page 2.) While Glass discloses "specialized function objects referred to as EJB function objects" (Glass, col. 15, lines 38-40), nothing in Glass teaches or suggests a "functional bean", for the following reasons. . .

Application No.: 09/620,102

Docket No.: 00-VE20.57

Glass is directed toward dynamically (*i.e.*, at run-time) generating proxy classes that support communications between client applications and server objects. (Glass, col. 4, lines 5-8.) Glass also teaches generating a number of other objects on both a client and a server to support client-server communications. (Glass, col. 4, lines 10-38.) Among these dynamically-generated objects are "type objects" for enabling access to the methods of a server object (Glass, col. 4, lines 7-8, 33-34), and "function objects", which provide a connection and correspond in number to the methods of a server object. (Glass, col. 4, lines 16-18, 34-38.) Among the functional objects disclosed by Glass are EJB function objects. Each EJB function object resides on the server and corresponds to a method in a server object. (Glass, Figs. 9 and 10.) The purpose of EJB function objects is to replace the need to create wrapper classes for each object on a server by providing "a common class, EJB function." (Glass, col. 15, lines 1-6.)

The above-described disclosure of Glass wholly fails to disclose any element corresponding to the recited "functional bean" in the context of the claimed invention. Function objects and EJB function objects do no more than "preliminary processing" common to many server objects before invoking a method in a server object. (Glass, col. 16, lines 8-15.) Glass contains no teaching or suggestion that EJB function objects contain any business logic. In fact, function objects, including EJB function objects, in corresponding to a single method of a server object, clearly teach away from functional beans that model an entire business function using a plurality of methods to implement business logic unique to that business function. (*See, e.g.*, Specification, page 15, lines 11-29.)

Further, Glass's "type object" contains the set of function objects that have a one-to-one correspondence with the methods in an associated server object. (Glass, col. 14, lines 29-33.) Type objects are in no way analogous to a "functional bean" because they fulfill no more than the function of forwarding messages to the appropriate function object, and clearly do not model or in any way represent business functions. (Glass, col. 16, lines 8-10.) Indeed, the purpose of Glass is to dynamically generate proxy classes and other classes to support client-server communications, but unlike Applicants' disclosure as reflected in claim 1, Glass contains no teaching or suggestion of altering the conventional Enterprise Java Beans architecture to model business functions in a functional bean accessible from a client as is required by Applicants' claim 1. Therefore, since Glass does not teach a functional bean, it cannot teach or suggest a *handle* to a functional bean and it therefore does not teach

Application No.: 09/620,102

Docket No.: 00-VE20.57

“returning to said client program a handle to a functional bean appropriate to the client request, wherein the functional bean is configured to model a business function,” as required by the claim.

Applicants have previously explained how their specification supports the novel feature of a “functional bean . . . configured to model a business function.” (See Paper No. 8.) Moreover, Glass is wholly silent with respect modeling business functions, and contains no teaching or suggestion of “returning a handle to a functional bean . . . configured to model a business function.” Accordingly, at least for the foregoing reasons, the 35 U.S.C. § 103(a) rejection of claim 1 as well as of claims 2-6 depending therefrom should be withdrawn and the claims allowed.

B. Claim 1: “a data store interface . . .” and “code providing for access to the data storage system . . .”

Claim 1 recites “a data store interface for coupling said application service program to a data storage system.” Claim 1 further recites “memory coupled to said application service program containing instructions executable by a processor . . . for servicing the queued customer requests in accordance with the code contained in the functional bean, said code providing for access to the data storage system via the data store interface.” The Examiner acknowledges that Glass does not teach these claim limitations. (Office Action, page 3). However, the Examiner contends that these limitations would have been obvious over Glass in view of Hightower and Pospisil. However, even assuming that Hightower and Pospisil taught the above-quoted claim limitations, which they do not and which is discussed below, neither reference contains a motivation for one of ordinary skill in the art to have combined the reference with the teachings of Glass, also discussed below.

1. Hightower

Hightower teaches “a method . . . for providing an application with intermittent connectivity support.” (Hightower, Abstract.) Accordingly, Hightower teaches storing method calls from a client to a server in a data store *on the client* until such time as there is a connection between the client and the server. (Hightower, col. 8, lines 27-42.) As the Examiner notes (Office Action, page 3), Hightower provides the benefit of allowing the user of a client application continuous use of the application “even in the absence of a connection to [an] enterprise application.” (Hightower, col. 8, lines 42-45.) However, Hightower contains no teaching or suggestion to use code in a functional bean providing access to a data storage system as is required by claim 1.

Application No.: 09/620,102

Docket No.: 00-VE20.57

a. Failure To Teach Each and Every Recited Claim Limitation

Initially, the Examiner's rejection of claim 1 as obvious over Glass in view of Hightower should be withdrawn because the references simply do not teach each and every recited claim limitation. In particular, contrary to the Examiner's assertion, Hightower's data store simply does not read on the "data storage system" recited in claim 1. Hightower's disclosed data store is resident on the client, whereas in the context of Applicants' claimed invention, both the recited "data store interface" and "data storage system" are *not* on the client. Rather, the data store interface exists for coupling an application service program, explicitly recited to be on a server, to a data storage system that is explicitly recited to be on a machine other than the client. Thus, Hightower plainly does not teach "memory coupled to said application service program, said memory for queuing customer requests and for servicing the queued customer requests in accordance with the code contained in the functional bean, said code providing for access to the data storage system via the data store interface" (emphasis added).

b. Inability To Combine Hightower and Glass

Because Hightower locates its data store on the client, any modification of Glass with the teachings of Hightower would result in a system with substantial structural differences from Applicants' claimed invention. Therefore, Glass and Hightower are incapable of combination to meet the limitations of claim 1, and cannot be used to reject claim 1 for this reason alone.

c. Lack of Motivation

Further, assuming *arguendo* that Glass discloses a functional bean, Hightower contains no disclosure that would have motivated one of ordinary skill in the art to modify Glass with a data storage interface and code in the functional bean "providing for access to the data storage system via the data store interface" as is required by claim 1. Hightower discloses at most the need to provide "applications with the ability to support intermittent connectivity processing." (Hightower, col. 2, lines 17-19.) Applicants, in contrast, address the problem of the depletion of server resources caused by the proliferation of server objects such as entity beans. (Specification, page 7, lines 3-14.) Hightower's solution to its disclosed need for supporting intermittent connectivity processing is to place a data store on client machines running applications requiring such support. Thus, both Hightower's disclosed problem, and Hightower's solution to that problem, have nothing to do with

Application No.: 09/620,102

Docket No.: 00-VE20.57

enabling access to a database server from a server object such as a functional bean. Accordingly, Hightower simply cannot contain any teaching or suggestion that would have motivated one of ordinary skill in the art to implement the recited "data store interface" or "functional bean [having] code providing for access to the data storage system via the data store interface."

2. Pospisil

Pospisil provides no motivation for one of ordinary skill in the art to have implemented "a data store interface for coupling said application service program to a data storage system" in the context of claim 1. The motivation cited by the Examiner (Office Action, page 3) is that "Pospisil would improve the system of Glass by providing database updating." However, Pospisil (page 3) teaches no more than that Enterprise Java Beans "execute . . . database updates using the standard JDBC API." Pospisil contains no teaching or suggestion of functional beans, much less of using functional beans to provide for access to a data storage system via a data store interface. The mere fact that Pospisil discusses database transactions such as updating in no way would have provided motivation to use functional beans to provide for access to a data storage system via a data storage interface. Indeed, Applicants' claimed invention addresses not the need to provide for database transactions such as updating, but rather to provide for such transactions in a more efficient way, *i.e.*, by using functional beans. Nothing in Pospisil suggests any problem or need arising from Enterprise Java Beans as known in the prior art that would have motivated one of ordinary skill to have implemented functional beans providing for access to a data storage system as recited in claim 1. Furthermore, Pospisil does not cure the deficiencies noted with respect to Glass and Hightower.

At least for the foregoing reasons, claim 1, as well as claims 2-6 depending therefrom, is patentable over the combination of Glass, Hightower, and Pospisil.

C. Applicants' Dependent Claims Include Patentable Subject Matter.

Applicants' dependent claims are separately patentable. For example, claim 4 recites that "the functional bean is configured to provide transactional persistence to a client transaction." The Examiner asserts that Glass teaches this claim limitation. However, the cited portion of Glass discloses only that the EJB function object provides "preliminary processing" including "transaction management." (Glass, col. 16, lines 10-15.) Glass says nothing at all about transactional *persistence*. Applicants' Specification (page 10, lines 21-

Application No.: 09/620,102

Docket No.: 00-VE20.57

22) explains that "transactional persistence" means that "the data objects operated on by a functional bean's methods and any state information are persisted." Glass contains absolutely no discussion of persisting either data objects or state information of any kind. Accordingly, Glass does not teach or suggest that "the functional bean is configured to provide transactional persistence." Therefore, for at least the foregoing additional reason, claim 4 is in condition for allowance.

II. REJECTION OF CLAIMS 7-19 AS OBVIOUS OVER GLASS IN VIEW OF POSPISIL

Claims 7-19 are rejected over Glass in view of Pospisil. However, the Examiner has failed to state a *prima facie* case of obviousness at least because (1) Glass and Pospisil, even when combined, failed to teach or suggest each and every one of the limitations of Applicants' claims, and (2) there is no teaching or suggesting in the prior art references to combine them to achieve the claimed invention. See MPEP § 2143. Accordingly, Applicants respectfully traverse the rejections of claim 1-6 for the independent reasons set forth below.

A. Claims 7 and 10: "functional beans"

Claim 7 recites *interalia* "a plurality of sets of functional beans, each set comprising at least one functional bean assigned to perform a particular business method" and also recites that "the client is configured to use the handle to interact with the functional bean to execute a business method." Claim 10 recites *interalia* a "functional bean, said functional bean comprising code to execute a particular business function." Accordingly, claims 7 and 10 are in condition for allowance at least for the reasons stated above with respect to claim 1 regarding Glass' failure to teach or suggest functional beans. Further, claims 8-9, depending from claim 7, and claims 11-17, depending from claim 10, are also in condition for allowance for at least the reasons stated above.

B. Claims 7, 12 and 18: "a load-sharing program . . .", "instructions to create a number of instances of functional beans . . .", and "provid[ing] transactional access to a pool of scarce system resources"

Claim 7 recites "a load-sharing program coupled to the service manager program and configured to create instances of functional beans based on a criterion." Claim 12 recites the code of a functional bean "further comprising instructions to create a number of instances of functional beans of the particular type requested, said number being dependent on available of resources." Claim 18 recites instructions in a functional bean that "provide transactional access to a pool of scarce system resources." The Examiner (Office Action, page 5)

Application No.: 09/620,102

Docket No.: 00-VE20.57

acknowledged that Glass is silent with respect to these claim limitations, but asserted that they would have been obvious in view of Pospisil because “the teaching of Pospisil would improve the system of Glass by optimizing resource usage.” (Office Action, pages 6, 7, 8.)

In fact, Pospisil (page 7) teaches no more than that “[s]calability is one of the most important performance aspects” of an Enterprise Java Beans application. Accordingly, Pospisil teaches a number of factors to test when evaluating an application’s performance. However, Pospisil contains no teaching or suggestion of creating functional beans – or any kind of object, for that matter – based on a criterion. Moreover, Pospisil contains no disclosure that would have motivated one of ordinary skill in the art to implement a load-sharing program as is required by claim 7, instructions regarding the number of instances of functional beans created as is required by claim 12, or instructions providing transactional access to a pool of scarce system resources as is required by claim 18.

Therefore, claims 7, 12, and 18 are in condition for allowance for at least two independent reasons. First, the cited references, including Glass and Pospisil, fail to teach all of the limitations of claims 7, 12, and 18. Second, even if Pospisil did teach these limitations, the cited references contain no motivation to modify Glass with the recited load-sharing program or instructions in a functional bean. Further, claims 8-9, depending from claim 7, and claim 19, depending from claim 18, are also in condition for at least each of these independent reasons.

C. Claim 18: “deriving a class with no data elements . . .”

Claim 18 recites in part “deriving a class with no data elements from the object-oriented middleware component.” The Examiner asserts that Glass teaches this claim limitation. (Office Action, page 8.) However, Glass contains absolutely no teaching or suggestion of “deriving a class” as is required by claim 18. Glass teaches merely that “[u]nique functionality may be added to each EJBfunction object after it has been instantiated to provide for unique processing needs included in function object.” The Examiner may be relying on this statement to meet the limitation in claim 18 of “adding a set of computer-executable instructions to the derived class”, in which case Glass cannot also be disclosing “deriving a class” as is separately recited in claim 18. In any event, Glass contains absolutely no disclosure of what this unique functionality is or how it is added, and therefore cannot be said to teach or suggest “deriving a class.”

Application No.: 09/620,102

Docket No.: 00-VE20.57

Further, Glass contains absolutely no teaching or suggestion that its EJB function object possesses no data elements and therefore Glass cannot read on "deriving a class with no data elements" for this independent reason.

For at least the foregoing reasons, claim 18 is in condition for allowance, as is claim 19, depending therefrom.

CONCLUSION

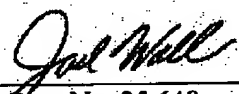
Applicants respectfully submit that all pending claims are distinguished over the cited prior art and are in condition for allowance. If the Examiner has any questions or issues relating to Applicants' response, he is encouraged to telephone the undersigned representative.

Any fees associated with the filing of this paper should be identified in an accompanying transmittal. However, if any additional fees are required in connection with the filing of this paper, permission is given to charge Deposit Account No. 07-2347, under order number 00-VE20.57. To the extent necessary, a further petition for extension of time under 37 C.F.R. § 1.136 is hereby made, the fee for which should be charged to the foregoing deposit account number.

Respectfully submitted,

Date: September 7, 2004

By:


Joel Wall, Reg. No. 25,648
Attorney for Applicants
Verizon Corporate Services Group, Inc.
c/o Christian Andersen
600 Hidden Ridge
Mailcode HQE03H14
Irving, TX 75038
972-718-4800
Customer No. 32127